A1 cont'd

the memory device to replace the defective memory locations. The RA solution is then translated into bank and fuse addresses corresponding to the antifuses that should be programmed to remap defective memory locations to the redundancy memory that was calculated by the RA solution at a step 104. The bank and fuse addresses are stored programmatically in a fuse address array. Typically, the fuse address array is stored by the tester performing the testing. Conceptually, along one axis of the fuse address array is a list of the fuse identification (ID) of every possible antifuse on a memory device, and along a second axis is every memory device for a group of devices. The group of devices may be all of the memory devices of a single wafer, or all of the memory devices of a single lot of wafers. As a result of storing the data in the fuse address array, the particular antifuses that need to be programmed for each of the memory devices of the group is available for reference.--

Please replace the paragraph beginning at page 2, line 27, with the following rewritten paragraph:

A2

--In going through the process of programming the antifuses of the devices of each touchdown as shown in Figure 1 at steps 106-110, the tester sequences through the list of the fuse IDs of the fuse address array. For each fuse ID, the tester evaluates the stored information to determine if any of the devices in the touchdown need the antifuse corresponding to the current fuse ID to be programmed. If none of the devices require programming of the antifuse, then the tester continues to the next fuse ID. However, at a step 112, if any one of the devices in the touchdown need the antifuse corresponding to the current fuse ID to be programmed, then the SVM for each DUT having a device in need of antifuse programming is programmed with a data value indicative of the bank address of the antifuse that needs to be programmed. For those devices in the touchdown that do not need the antifuse corresponding to the current fuse ID to be programmed, a null value is provided to the respective data pins instead.--

Please replace the paragraph beginning at page 3, line 10, with the following rewritten paragraph:

--The fuse address of the antifuse to be programmed is commonly applied to the address terminals of each of the DUTs as in step 114. For those devices receiving a null data value, the fuse address has no effect. However, for the devices also receiving a data value indicative of a particular fuse bank, the application of the fuse address triggers a fuse blow event that programs the antifuse corresponding to the bank address applied to the data pins of the device and the fuse address applied to the address pins of the device as in step 116. The fuse address must be held valid throughout the entire duration of the fuse blow event, otherwise, the fuse blow event is terminated and the antifuse will not be programmed.--

A3

Please replace the paragraph beginning at page 3, line 18, with the following rewritten paragraph:

--Upon completion of the fuse blow event, at a step 118, the tester continues to the next fuse ID of the fuse address array, and the process of determining whether any of the devices in the touchdown require programming of the antifuse of the current fuse ID, and programming thereof if at least one device requires it, continues until all of the possible antifuses of the memory device are checked.--

A4

Please replace the paragraph beginning at page 6, line 1, with the following rewritten paragraph:

--Figure 2 illustrates a portion of a memory device 300 according to an embodiment of the present invention. The memory device 300 is similar to the conventional memory device, and such conventional elements have not been shown in order to avoid obscuring the present invention. For example, the memory device 300 includes a memory array (not shown) having conventional redundant memory used to replace defective memory cells of the array. As with a conventional memory device, remapping of defective memory cells to the redundant memory is accomplished by programming a programmable device, such as an antifuse. Conventional antifuse programming circuitry 338 is used for programming of the

A5

antifuses. In the present embodiment, the memory array is divided into four array regions, each of which has a fuse bank region associated therewith. The four fuse bank regions 320a, 320b, 320c, and 320d are shown in Figure 2. Within each of the fuse bank regions are antifuses that can be programmed to replace defective memory cells of the associated memory array with the redundant memory. The antifuses can be uniquely identified by a bank address and a fuse address, and when programming of the antifuses takes place, the bank and fuse address are used to identify the antifuse that needs to be programmed. Programming of the antifuses in the four fuse bank regions 320a-320d is conventional and known to those ordinarily skilled in the art.--

A5 Concl'd

Please replace the paragraph beginning at page 6, line 13, with the following rewritten paragraph:

--Although the memory device 300 includes many aspects that are conventional, the memory device 300 is different in that it includes bank address latches 330, 332, and fuse address latches 340, 342, 344, 346, and logic circuitry 350 to enable parallel antifuse programming according to embodiments of the present invention. The bank address latches 330, 332, and the fuse address latches 340, 342, 344, and 346, are coupled to receive address signals A0-An from the external address terminals of the memory device 300. The logic circuitry 350 is coupled to external data terminals DQ0-DQ3, to receive command signals, and is further coupled to the bank address latches 330, 332 and the fuse address latches 340-346. As will be explained in more detail below, when the memory device 300 is in an antifuse programming mode, command signals can applied to the memory device 300 via data terminals DQ0-DQ3 to instruct the loading of an address applied to the address terminals A0-An into either the bank address latches 330, 332, or the fuse address latches 340, 342, 344, 346. The logic circuitry 350 receives the command signals on DQ0-DQ3, and in response, generates BANKLAT0 and BANKLAT1 signals to instruct the bank latches 330, 332 to latch an antifuse bank address from the external address terminals, and generates signals CNTRL0-CNTRL3 to instruct the fuse address latches 340-346 to latch an antifuse address from the external address terminals. The latched bank and antifuse addresses are then provided to the respective banks 320a-320d to uniquely select an

A6

A6 concl'd

antifuse for programming. As shown in Figure 2, conventional antifuse programming circuitry 338 is included for programming antifuses of the antifuse banks 320a-320d, as is well known.--

Please replace the paragraph beginning at page 6, line 22, with the following rewritten paragraph:

A7

--It will be appreciated that the memory device 300 includes additional conventional circuitry that has not been shown in Figure 2. These circuits are well known in the art, and consequently, will not be discussed herein for the sake of brevity. However, the description provided herein is sufficient to enable those of ordinary skill in the art to practice the invention. An alternative memory device suitable for use in embodiments of the present invention is described in more detail in co-pending U.S. Patent Application No. 09/810,366 to Cowles, entitled CIRCUIT AND METHOD FOR TEST AND REPAIR and filed March 15, 2001, which is incorporated herein by reference.--

Please replace the paragraph beginning at page 7, line 28, with the following rewritten paragraph:

A8

--Figure 3 illustrates a flow diagram of an embodiment of the parallel antifuse programming process of the present invention. As with the conventional process, at a step 352, an RA solution is calculated based on the location of the defective memory cells and then at a step 354, translated into bank and fuse addresses corresponding to the antifuses that need to be programmed for each of the devices in a group. As previously mentioned, the group may be the devices on a single wafer, or the devices in a single lot of wafers. The bank and fuse address are stored in a data array as in the conventional process. However, in contrast with the conventional antifuse programming process, following the translation of the RA solution, the bank and fuse addresses of the antifuses for a first region of redundant memory of each device in a touchdown are loaded into a tester scramble memory SCRAM. In the present example that the antifuses for each device are programmed one fuse bank region at a time. However, it will be appreciated that the antifuse programming process can be performed for multiple regions at a time without departing from the scope of the present invention. Figure 5 illustrates a table 500 that provides an example of the organization of the bank and fuse address data 502 and 504, respectively,

A8, Cont'd

stored in the tester SCRAM. Null values 506 are inserted into the bank and fuse addresses for the memory devices in the touchdown that do not require as many antifuses to be programmed. As will be explained in more detail below, this allows for the cycle of loading the bank and fuse addresses can be maintained for consistency during the fuse blow events. Although Figure 5 illustrates an example of a suitable arrangement for the bank and fuse addresses, as well as the appropriate load commands to apply to the DQs of the devices in the touchdown, it will be appreciated that other arrangements can be used as well without departing from the scope of the present invention.--

Please replace the paragraph beginning at page 8, line 18, with the following rewritten paragraph:

A9

--Following the loading of the bank and fuse addresses for the first fuse bank region into the tester SCRAM at a step 356, the process of having each memory device latch the respective bank and fuse address is carried out by programming the SVMs of each DUT in the touchdown to provide the appropriate load commands to the DQs of each of the memory devices. In the present example, loading of a respective bank and fuse address for each device of the touchdown is performed in sequence, with the sequence being repeated until all of the antifuses in the particular region for the worst case device are programmed as in steps 358-366. The antifuse programming process is then repeated for each of the remaining fuse bank regions until all of the antifuses of the memory devices have been programmed as in steps 368-372.--

Please replace the paragraph beginning at page 11, line 1, with the following rewritten paragraph:

A10

--An alternative embodiment of the present invention will be described with respect to Figures 6 and 7, and is directed to a tester having the capability of providing each device of a touchdown with unique address signals. This allows for the respective bank address and fuse address of each device to be loaded concurrently, thus, saving even more time than the previously described embodiment.--

Please replace the paragraph beginning at page 11, line 6, with the following rewritten paragraph:

A11

--Figure 6 illustrates a flow diagram of a parallel antifuse programming process according to an alternative embodiment of the present invention. An RA solution is calculated at a step 602 based on the location of the defective memory cells and then translated into bank and fuse addresses corresponding to the antifuses that need to be programmed for each of the devices in a group at a step 604. The bank and fuse address are stored in a data array as in the conventional process. The tester then applies the appropriate signals to enable the antifuse programming mode for all the devices in a touchdown.--

Please replace the paragraph beginning at page 11, line 20, with the following rewritten paragraph:

A12

--Following the entry into the antifuse programming mode, the bank and fuse addresses for the first antifuse in need of programming in each of the devices of the touchdown is loaded into the respective data registers for each of the DUTs at steps 606 and 608. As with the previous embodiment, the antifuses are programmed in a per fuse bank region basis. The timing diagram of Figure 7 illustrates programming the antifuses of fuse bank region 0. A 1H value is applied to the DQs of all of the memory devices in the touchdown to initiate a Load Bank Address Bank 0 and 1 command, and at a time T0, each of the memory devices in the touchdown latches in the bank address applied to the respective address terminals. The tester then switches the address signals to provide the respective fuse address to each of the memory devices and changes the data value applied to the DQs to 5H to initiate a Load Fuse Address Bank 0 command. At a time T1, the respective fuse addresses are latched by each memory device and a fuse blow event is triggered for each of the memory devices as shown in Figure 6 at step 610. While the fuse blow event is occurring, the bank and fuse addresses for next fuse in the region in need of programming for each memory device is programmed into the respective data registers. By programming the bank and fuse addresses for the next antifuse during the fuse blow event for the previous antifuse, the operation can essentially be hidden.--

Please replace the paragraph beginning at page 12, line 18, with the following rewritten paragraph:

A13

--The process of loading the bank and fuse addresses into the respective data registers for the next antifuse to be programmed in the region for each individual device, latching the bank address, then latching the fuse address and triggering a fuse blow event to program the antifuse corresponding to the respective bank and fuse addresses in each device, is repeated until all of the fuses in that region for the worst case device are programmed as shown in Figure 6 in steps 606-612. As with the previously described embodiment, null values are provided to the devices having less antifuses in need of programming than the worst case memory device and that are waiting for the programming of those antifuses to be completed.--

Please replace the paragraph beginning at page 12, line 26, with the following rewritten paragraph:

A14

--When programming of the antifuses for a region is completed, the process is repeated for the next region, and all remaining regions until all of the antifuses in need of programming for each device in the touchdown are programmed as shown in steps 606-614.--

In the Claims:

Please cancel claims 28, 29, 34, and 35.

Please amend claims 26 and 32, as follows:

A15

26. (Once amended) A memory device having external address terminals and data terminals, and further having an array of memory with redundant memory to replace memory cells therein in accordance with programmed programmable elements, the memory device comprising:

5

an antifuse bank address latch coupled to the address terminals for latching an antifuse bank address applied to the address terminals corresponding to a bank of antifuses including a programmable element to be programmed by a programming event;

7